Ground state of the time-independent Gross-Pitaevskii equation

Claude M. Dion [a], Eric Cancès [b]

[a] *Department of Physics, Umeå University, SE-90187 Umeå, Sweden*

[b] *CERMICS, École Nationale des Ponts et Chaussées and INRIA, 6 & 8, avenue Blaise Pascal, cité Descartes, F-77455 Marne-la-Vallée Cedex 2, France*

Abstract

We present a suite of programs to determine the ground state of the time-independent Gross-Pitaevskii equation, used in the simulation of Bose-Einstein condensates. The calculation is based on the Optimal Damping Algorithm, ensuring a fast convergence to the true ground state. Versions are given for the one-, two-, and three-dimensional equation, using either a spectral method, well suited for harmonic trapping potentials, or a spatial grid.

PACS: 03.75.Hh; 03.65.Ge; 02.60.Pn; 02.70.-c

Key words: Gross-Pitaevskii equation; Bose-Einstein condensate; ground state; Optimal Damping Algorithm.

PROGRAM SUMMARY

Manuscript Title: Ground state of the time-independent Gross-Pitaevskii equation
Authors: Claude M. Dion and Eric Cancès
Program Title: GPODA
Journal Reference:
Catalogue identifier:
Licensing provisions: none
Programming language: Fortran 90
Computer: any
Compilers under which the program has been tested: Absoft Pro Fortran, The Portland Group Fortran 90/95 compiler, Intel Fortran Compiler
RAM: From < 1 MB in 1D to $\sim 10^2$ MB for a large 3D grid
Keywords: Gross-Pitaevskii equation, Bose-Einstein condensate, Optimal Damping Algorithm

Email addresses: `claude.dion@tp.umu.se` (Claude M. Dion), `cances@cermics.enpc.fr` (Eric Cancès).

PACS: 03.75.Hh; 03.65.Ge; 02.60.Pn; 02.70.-c
Classification: 2.7 Wave Functions and Integrals, 4.9 Minimization and Fitting
External routines: External FFT or eigenvector routines may be required

Nature of problem:
The order parameter (or wave function) of a Bose-Einstein condensate (BEC) is obtained, in a mean field approximation, by the Gross-Pitaevskii equation (GPE) [1]. The GPE is a nonlinear Schrödinger-like equation, including here a confining potential. The stationary state of a BEC is obtained by finding the ground state of the time-independent GPE, *i.e.*, the order parameter that minimizes the energy. In addition to the standard three-dimensional GPE, tight traps can lead to effective two- or even one-dimensional BECs, so the 2D and 1D GPEs are also considered.
Solution method:
The ground state of the time-independent of the GPE is calculated using the Optimal Damping Algorithm [2]. Two sets of programs are given, using either a spectral representation of the order parameter [3], suitable for a (quasi) harmonic trapping potential, or by discretizing the order parameter on a spatial grid.
Running time:
From seconds in 1D to a few hours for large 3D grids.
References:

[1] F. Dalfovo, S. Giorgini, L. P. Pitaevskii, S. Stringari, Rev. Mod. Phys. 71 (1999) 463.

[2] E. Cancès, C. Le Bris, Int. J. Quantum Chem. 79 (2000) 82.

[3] C. M. Dion, E. Cancès, Phys. Rev. E 67 (2003) 046706.

LONG WRITE-UP

1 Introduction

Advances in cooling methods for dilute atomic gases have made it possible to attain a new state of matter, the Bose-Einstein condensate (BEC) [1,2]. As the temperature of atoms gets very low, their de Broglie wavelength, an inherently quantum character, can become greater than the interatomic distance. At that point, bosonic atoms will "condense" into a unique quantum state and become indistinguishable parts a of macroscopic quantum object, the BEC. It has now been achieved for all stable alkali atoms [3,4,5,6,7], as well as with hydrogen [8], metastable helium [9,10], and for diatomic molecules [11].

Starting from the many-body Hamiltonian describing the cold atoms, it is possible to reduce the problem, by considering the order parameter, or wave function, for the condensed fraction only. It is governed by a nonlinear Schrödinger equation, the Gross-Pitaevskii equation (GPE) [12,13,14,15,16]

$$\left[-\frac{\hbar^2}{2m}\boldsymbol{\nabla}_{\mathbf{X}}^2 + V_{\text{trap}}(\mathbf{x}) + \lambda_{\text{3D}}\left|\psi(\mathbf{x})\right|^2\right]\psi(\mathbf{x}) = \mu\psi(\mathbf{x}), \tag{1}$$

with the normalization condition $\|\psi\|_{L^2} = 1$, where $\hbar$ is the reduced Planck constant, m the mass of the boson, V_{trap} a trapping potential spatially confining the condensate, and μ the chemical potential of the condensate. Physically, the nonlinearity corresponds to the mean field exerted on one boson by all the others and is given, for a condensate of N bosons in 3D, by

$$\lambda_{\text{3D}} \equiv g_{\text{3D}}N = \frac{4\pi\hbar^2 aN}{m}. \tag{2}$$

The value of a, the scattering length, varies according to the species of bosons being considered. The energy associated with the wave function $\psi(\mathbf{x})$ is obtained according to [12,13,14,15,16]

$$E[\psi] = N\int_{\mathbb{R}^3}\left[\frac{\hbar^2}{2m}\left|\boldsymbol{\nabla}\psi(\mathbf{x})\right|^2 + V_{\text{trap}}(\mathbf{x})\left|\psi(\mathbf{x})\right|^2 + \frac{\lambda_{\text{3D}}}{2}\left|\psi(\mathbf{x})\right|^4\right]\mathrm{d}\mathbf{x}. \tag{3}$$

We present here a suite of programs designed to calculate the ground state of the GPE, *i.e.*, the order parameter $\psi(\mathbf{x})$ with to the lowest energy. This corresponds to the actual condensate order parameter, in the absence of any excitation. The problem is thus to find the ground state of the condensate, that is a normalized function $\psi_{\text{GS}}(\mathbf{x})$ that minimizes $E[\psi]$. Recall that if V_{trap} is continuous and goes to $+\infty$ at infinity, and if $\lambda_{\text{3D}} \geq 0$, the ground state of $E[\psi]$

exists and is unique up to a global phase. In addition, the global phase can be chosen such that $\psi_{\rm GS}$ is real-valued, and positive on $\mathbb{R}^3$. The ground state $\psi_{\rm GS}$ can be computed using the Optimal Damping Algorithm (ODA), originally developed for solving the Hartree-Fock equations [17,18]. This algorithm is garanteed to converge to the ground state. Two different discretizations of the order parameter are available in our sets of programs. In one case, a basis set of eigenfunctions of the harmonic oscillator is used, which is particularly suited for a harmonic (or quasi-harmonic) trapping potential $V_{\rm trap}$. In this case, an efficient method to convert from the spectral representation to a spatial grid [19] is employed to treat the nonlinearity. In the other case, a spatial grid is used throughout, with the kinetic energy derivative evaluated with the help of Fast Fourier Transforms. Note that, in all cases, the value of the energy given on output is actually the energy per particle, $E[\psi]/N$.

2 Optimal Damping Algorithm

To describe the ODA [17,18] in the context of the GPE, we start by defining the operators

$$\hat{H}_0 \equiv -\frac{\hbar^2}{2m}\nabla^2_{\mathbf{x}} + V(\mathbf{x}), \tag{4}$$

corresponding to the linear part of the GPE (1), and

$$\hat{H}(\rho) \equiv \hat{H}_0 + \lambda_{\rm 3D}\rho(\mathbf{x}) \tag{5}$$

the full, nonlinear Hamiltonian, where we have introduced $\rho \equiv |\psi|^2$ ($N\,\rho(\mathbf{x})$ is the density of the condensate at point $\mathbf{x}$).

The ODA is based on the fact that the ground state density matrix $\gamma_{\rm GS} = |\psi_{\rm GS}\rangle\langle\psi_{\rm GS}|$ is the unique minimizer of

$$\inf\left\{\mathcal{E}[\gamma],\ \gamma \in \mathcal{S}(L^2(\mathbb{R}^3)),\ 0 \le \gamma \le I,\ \mathrm{tr}(\gamma) = 1\right\}. \tag{6}$$

In the above minimization problem, $\mathcal{S}(L^2(\mathbb{R}^3))$ denotes the vector space of bounded self-adjoint operators on $L^2(\mathbb{R}^3)$ and I the identity operator on $L^2(\mathbb{R}^3)$. The energy functional $\mathcal{E}[\gamma]$ is defined by

$$\mathcal{E}[\gamma] = \mathrm{tr}(\hat{H}_0\gamma) + \frac{\lambda_{\rm 3D}}{2}\int_{\mathbb{R}^3}\rho_\gamma^2,$$

where $\rho_\gamma(\mathbf{x}) = \gamma(\mathbf{x},\mathbf{x})$ ($\gamma(\mathbf{x},\mathbf{y})$ being the kernel of the trace-class operator γ). The ODA implicitly generates a minimizing sequence γ_k for (6), starting, for instance, from the initial guess $\gamma_0 = |\psi_0\rangle\langle\psi_0|$, where ψ_0 is the ground state of $\hat{H}_0$. The iterate γ_{k+1} is constructed from the previous iterate γ_k in two steps:

- Step 1: compute a normalized order parameter ψ'_k which minimizes

$$s_k = \inf \left\{ \frac{\mathrm{d}}{\mathrm{d}t} \mathcal{E}\left[(1-t)\gamma_k + t|\psi\rangle\langle\psi|\right]\Big|_{t=0}, \quad \|\psi\|_{L^2} = 1 \right\}.$$

  It is easy to check that ψ'_k is in fact the ground state of $\hat{H}(\rho_{\gamma_k})$ and that either $\psi'_k = \psi_{\mathrm{GS}}$ (up to a global phase) or $s_k < 0$.
- Step 2: compute

$$\alpha_k = \operatorname{arginf}\left\{\mathcal{E}\left[(1-t)\gamma_k + t|\psi'_k\rangle\langle\psi'_k|\right], \quad t \in [0,1]\right\}$$

  and set $\gamma_{k+1} = (1-\alpha_k)\gamma_k + \alpha_k|\psi'_k\rangle\langle\psi'_k|$. Note that α can be computed analytically, for the function $t \mapsto \mathcal{E}\left[(1-t)\gamma_k + t|\psi'_k\rangle\langle\psi'_k|\right]$ is a second order polynomial of the form $\mathcal{E}[\gamma_k] + ts_k + \frac{t^2}{2}c_k$.

The set

$$\mathcal{C} = \left\{\gamma \in \mathcal{S}(L^2(\mathbb{R}^3)),\ 0 \leq \gamma \leq I,\ \mathrm{tr}(\gamma) = 1\right\}$$

being convex, $\gamma_k \in \mathcal{C}$ for all k and either $\gamma_k = \gamma_{\mathrm{GS}}$ or $\mathcal{E}[\gamma_{k+1}] < \mathcal{E}[\gamma_k]$. In addition, it can be proved that, up to a global phase, ψ'_k converges to ψ_{GS} when k goes to infinity. Likewise, $\rho_k \equiv \rho_{\gamma_k}$ converges to $\rho_{\mathrm{GS}} \equiv \psi^2_{\mathrm{GS}}$. It is important to note that the sequences ψ'_k and ρ_k can be generated without explicitely computing γ_k. This is crucial to reduce the overall memory requirement of ODA.

Let us now describe a practical implementation of ODA, in which only order parameters and densities are stored in memory. The algorithm is initialized by ψ_0, from which we derive $\rho_0 = |\psi_0|^2$, $f_0 = (\psi_0, \hat{H}_0\psi_0)$, and $h_0 = (\psi_0, \hat{H}(\rho_0)\psi_0)$. The iterations go as follows:

(1) Calculate the ground state ψ'_k of $\hat{H}(\rho_k)$, and $\rho'_k = |\psi'_k|^2$.
(2) Compute

$$\begin{aligned} f'_k &= (\psi'_k, \hat{H}_0\psi'_k), \\ h'_k &= (\psi'_k, \hat{H}(\rho_k)\psi'_k), \\ h''_k &= (\psi'_k, \hat{H}(\rho'_k)\psi'_k). \end{aligned}$$

(3) Calculate

$$\begin{aligned} s_k &= h'_k - h_k, \\ c_k &= h_k + h''_k - 2h'_k + f'_k - f_k. \end{aligned}$$

(4) Set $\alpha_k = 1$ if $c_k \leq -s_k$, $\alpha_k = -s_k/c_k$ otherwise, and

$$
\begin{aligned}
E_{\text{opt}} &= \frac{1}{2}\left(f_k + h_k\right) + \alpha_k s_k + \frac{\alpha_k^2}{2} c_k, \\
\rho_{k+1} &= (1-\alpha_k)\rho_k + \alpha_k \rho_k', \\
f_{k+1} &= (1-\alpha_k) f_k + \alpha_k f_k', \\
h_{k+1} &= 2E_{\text{opt}} - f_{k+1}.
\end{aligned}
$$

(5) If $|s_k/E_{\text{opt}}| > \varepsilon_{\text{ODA}}$ (convergence criterion), go to (1), otherwise compute the ground state of $H(\rho_{k+1})$, which is the solution sought, and terminate.

To calculate the ground state of the operators $\hat{H}_0$ and $\hat{H}(\rho)$, the inverse power method is used, with the convergence criterion $|E_{i+1} - E_i| \leq \varepsilon_{\text{IP}}$, where E are the lowest eigenvalues at consecutive iterations. The inverse power algorithm itself uses the conjugated gradient method to solve $\hat{H}v = u$, with u given and v unknown. The convergence of the conjugated gradient is controlled by the criterion ε_{CG}. The only exception to this is in `gpoda1Ds`, where the ground states of the operators are found by a matrix eigenproblem solver routine (see Sec. 4.6.1).

3 Representations of the GPE

The Gross-Pitaevskii equation was defined in Eq. (1), with the nonlinearity Eq. (2) in 3D. In this work, we are also considering cases where the confinement V_{trap} is so tight in some spatial dimension that the condensate can actually be considered as a two-, or even one-dimensional object. This leads to different representations of the nonlinearity λ and the expression for the coupling parameters g_{2D} and g_{1D} can be found in Refs. [20,21,22]. We refer to chapter 17 of [2] for a detailed discussion of the validity of the mean field approximation in these cases.

3.1 Spatial grid approach

If the order parameter is represented on a discretized spatial grid, the calculation of the potential energy and the nonlinearity are trivial, as they both act locally, while the kinetic energy operator is non-local. By means of a Fourier transform, it is possible to convert from position to momentum space, where the kinetic operator is local. This is implemented by means of a Fast Fourier Transforms (FFTs), allowing to convert back and forth between the two representations, to evaluate each part of the Hamiltonian in the space where it is local.

3.2 Spectral method

For many situations, the trapping potential is harmonic, or a close variation thereof, *i.e.*,

$$V_{\text{trap}}(x,y,z) = \frac{m}{2}\left(\omega_x^2 x^2 + \omega_y^2 y^2 + \omega_z^2 z^2\right) + V_0(x,y,z), \tag{7}$$

where ω is the trapping frequencies in each direction and V_0 accounts for eventual corrections to a purely harmonic trap. In this case, it is advantageous to use a basis set made up of the eigenfunctions of the quantum harmonic oscillator.

We start by rescaling Eq. (1), introducing dimensionless lengths $(\tilde{x},\tilde{y},\tilde{z})$,

$$x = \left(\frac{\hbar}{m\omega_x}\right)^{1/2}\tilde{x}, \tag{8a}$$

$$y = \left(\frac{\hbar}{m\omega_y}\right)^{1/2}\tilde{y}, \tag{8b}$$

$$z = \left(\frac{\hbar}{m\omega_z}\right)^{1/2}\tilde{z}, \tag{8c}$$

and a new order parameter $\tilde{\psi}$ defined as

$$\psi(x,y,z) = A\tilde{\psi}(x,y,z). \tag{9}$$

Considering the normalization condition

$$\int_{\mathbb{R}^3} |\psi(x,y,z)|^2 \,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z = 1, \tag{10}$$

we take

$$A = \left(\frac{m}{\hbar}\right)^{3/4}(\omega_x\omega_y\omega_z)^{1/4} \tag{11}$$

such that

$$\int_{\mathbb{R}^3} \left|\tilde{\psi}(\tilde{x},\tilde{y},\tilde{z})\right|^2 \mathrm{d}\tilde{x}\,\mathrm{d}\tilde{y}\,\mathrm{d}\tilde{z} = 1. \tag{12}$$

The Gross-Pitaevskii equation now reads

$$\left[\frac{\omega_x}{\omega_z}\left(-\frac{1}{2}\nabla_{\tilde{x}}^2 + \frac{\tilde{x}^2}{2}\right) + \frac{\omega_y}{\omega_z}\left(-\frac{1}{2}\nabla_{\tilde{y}}^2 + \frac{\tilde{y}^2}{2}\right) + \left(-\frac{1}{2}\nabla_{\tilde{z}}^2 + \frac{\tilde{z}^2}{2}\right) + \tilde{V}_0(\tilde{x},\tilde{y},\tilde{z}) + \tilde{\lambda}_{3\text{D}}\left|\tilde{\psi}(\tilde{x},\tilde{y},\tilde{z})\right|^2\right] = \tilde{\mu}\tilde{\psi}(\tilde{x},\tilde{y},\tilde{z}), \tag{13}$$

with

$$\tilde{V}_0(\tilde{x},\tilde{y},\tilde{z}) \equiv \frac{1}{\hbar\omega_z}V_0(x,y,z), \tag{14}$$

$$\tilde{\lambda}_{3\mathrm{D}} \equiv \frac{m^{3/2}}{\hbar^{5/2}} \left(\frac{\omega_x \omega_y}{\omega_z}\right)^{1/2} g_{3\mathrm{D}} N = 4\pi a N \left(\frac{m}{\hbar}\frac{\omega_x \omega_y}{\omega_z}\right)^{1/2}, \tag{15}$$

and

$$\tilde{\mu} \equiv \frac{\mu}{\hbar\omega_z}. \tag{16}$$

Similarly,

$$\tilde{E}[\tilde{\psi}] \equiv \frac{E[\psi]}{\hbar\omega_z}. \tag{17}$$

Using the Galerkin approximation, we can express the order parameter $\tilde{\psi}$ as a linear combination of a finite number of (orthonormal) basis functions ϕ,

$$\tilde{\psi}(\tilde{x}, \tilde{y}, \tilde{z}) = \sum_{i=0}^{N_{\tilde{x}}} \sum_{j=0}^{N_{\tilde{y}}} \sum_{k=0}^{N_{\tilde{z}}} c_{ijk} \phi_i(\tilde{x}) \phi_j(\tilde{y}) \phi_k(\tilde{z}), \tag{18}$$

where the ϕ are chosen as the eigenfunctions of the 1D harmonic oscillator, *i.e.*,

$$\left(-\frac{1}{2}\frac{\mathrm{d}^2}{\mathrm{d}\xi^2} + \frac{\xi^2}{2}\right) \phi_n(\xi) = \left(n + \frac{1}{2}\right) \phi_n(\xi). \tag{19}$$

In the spectral representation of Eq. (18), Eq. (13) becomes a series of coupled equations for the coefficients c_{ijk}, and the first part of the Hamiltonian can be evaluated by a simple multiplication, according to Eq. (19). The second part of the Hamiltonian, consisting of the $\tilde{V}_0$ and the nonlinear terms, is local in $(\tilde{x}, \tilde{y}, \tilde{z})$ and couples the different coefficients. Its operation can be calculated in a manner similar to what is used for the spatial grid (see Sec. 3.1): starting from the coefficients c_{ijk}, the order parameter $\tilde{\psi}$ is evaluated at selected grid points $(\tilde{x}, \tilde{y}, \tilde{z})$, the local terms are then trivially calculated, and the order parameter is transformed back to the spectral representation. This procedure can be performed efficiently and accurately using the method described in Ref. [19].

For the 2D case, *i.e.*, when the motion along y is suppressed, we rescale the lengths according to Eq. (8), which results in

$$A = \left(\frac{m}{\hbar}\right)^{1/2} (\omega_x \omega_z)^{1/4} \tag{20}$$

for the scaling factor of the order parameter. We thus obtain the 2D GPE

$$\left[\frac{\omega_x}{\omega_z}\left(-\frac{1}{2}\nabla_{\tilde{x}}^2 + \frac{\tilde{x}^2}{2}\right) + \left(-\frac{1}{2}\nabla_{\tilde{z}}^2 + \frac{\tilde{z}^2}{2}\right) + \tilde{V}_0(\tilde{x}, \tilde{z}) + \tilde{\lambda}_{2\mathrm{D}} \left|\tilde{\psi}(\tilde{x}, \tilde{z})\right|^2\right] = \tilde{\mu}\tilde{\psi}(\tilde{x}, \tilde{z}), \tag{21}$$

where

$$\tilde{\lambda}_{2\mathrm{D}} \equiv \lambda_{2\mathrm{D}} \frac{m}{\hbar^2} \left(\frac{\omega_x}{\omega_z}\right)^{1/2}. \tag{22}$$

Similarly, we get for the one-dimensional case (where the motion along x and y is frozen)

$$A = \left(\frac{m\omega_z}{\hbar}\right)^{1/4}, \tag{23}$$

$$\left[-\frac{1}{2}\nabla_{\tilde{z}}^2 + \frac{\tilde{z}^2}{2} + \tilde{V}_0(\tilde{z}) + \tilde{\lambda}_{1\mathrm{D}}\left|\tilde{\psi}(\tilde{z})\right|^2\right] = \tilde{\mu}\tilde{\psi}(\tilde{z}), \tag{24}$$

and

$$\tilde{\lambda}_{1\mathrm{D}} \equiv \lambda_{1\mathrm{D}}\left(\frac{m}{\hbar^3\omega_z}\right)^{1/2}. \tag{25}$$

4 Description of the programs

4.1 `gpoda3Dg`

This program solves the full 3D GPE (1) on a grid. Atomic units are used throughout.

4.1.1 User-supplied routines

The double precision function `potentialV(x,y,z)` takes as input the three double precision arguments `x`, `y`, and `z`, corresponding to the spatial coordinates (x, y, z), and returns $V_{\mathrm{trap}}(x, y, z)$.

A 3D FFT routine must also be supplied. The program is set up to work with the DFFTPACK [23] transform of a real function, and can be linked directly to this library.

If the user wishes to use another FFT, the file `fourier3D.f90` must be modified accordingly. The program first calls `fft_init(n)`, where `n` is a one-dimensional integer array of length 4, the last three elements containing the number of grid points in x, y, and z, with the first element corresponding to the maximum number of grid points in any direction, *i.e.*, for `n(0:3)`, `n(0) = maxval(n(1:3))`. The program will then call repeatedly the subroutine `fourier3D(n,fin,fout,direction)`, with `fin` and `fout` double precision arrays of dimension `(n(1),n(2),n(3))`, and `direction` an integer. The routine should return in array `fout` the forward Fourier transform of `fin` if `direction` $= 1$, and the inverse transform for `direction` $= -1$. Any variable initialized by `fft_init` must be passed to `fourier3D` through a module. Note that the main program expects to receive the Fourier coefficients (following the forward transform) according to:

$$
\begin{aligned}
c_1 &= \sum_{n=1}^{N} f_n, \\
c_{2m-2} &= \sum_{n=1}^{N} f_n \cos\left[\frac{2\pi(m-1)(n-1)}{N}\right], \quad m = 2, \ldots, N/2+1 \\
c_{2m-1} &= -\sum_{n=1}^{N} f_n \sin\left[\frac{2\pi(m-1)(n-1)}{N}\right], \quad m = 2, \ldots, N/2
\end{aligned}
$$

where the coefficients c_m correspond to variable `fout` and the sequence f_n to `fin`.

4.1.2 Input parameters

The input parameters are read from a namelist contained in a file named `params3Dg.in`, with the following format (the variable type is indicated in parenthesis, where dp stands for double precision):

```
&params3Dg
 mass = mass of the boson (dp),
 lambda = nonlinearity λ_3D (dp),
 ng_x = number of grid points in x, (integer),
 ng_y = number of grid points in y, (integer),
 ng_z = number of grid points in z, (integer),
 xmin = first point of the grid in x (dp),
 xmax = last point of the grid in x (dp),
 ymin = first point of the grid in y (dp),
 ymax = last point of the grid in y (dp),
 zmin = first point of the grid in z (dp),
 zmax = last point of the grid in z (dp),
 critODA = convergence criterion for the ODA, ε_ODA (dp),
 critIP = convergence criterion for the inverse power, ε_IP (dp),
 critCG = convergence criterion for the conjugated gradient, ε_CG (dp),
 itMax = maximum number of iterations of the ODA (integer),
 guess_from_file = read initial guess from file guess3Dg.data? (logical)
&end
```

If the value of the input parameter `guess_from_file` is `.true.`, a file named `guess3Dg.data` must be present in the local directory. It contains the initial guess for the order parameter, and must consist in `ng_x` × `ng_y` × `ng_z` lines, each containing the values of the coordinates x, y, and z, followed by $\psi(x, y, z)$. *Note that the program does* ***not*** *check if the coordinates correspond to the grid defined by the input parameters.* The program will simply assign the first value of ψ to the first grid point, $(x_{\min}, y_{\min}, z_{\min})$, then the second value to the second grid point in x, with $y = y_{\min}$ and $z = z_{\min}$, etc. After n_x

points have been read, the next value of ψ is assigned to the second grid point in y, with $x = x_{\min}$ and $z = z_{\min}$, and so on. In other words, the fourth column of `guess3Dg.data` contains $\psi(x, y, z)$ in standard Fortran format, with x corresponding to the first index, y to the second, and z to the third.

4.1.3 Output files

The order parameter is written out in file `gs3Dg.data`, with each line containing the coordinates x, y, and z, followed by $\psi(x, y, z)$. If the algorithm has not converged, the file will contain the function obtained at the last iteration. The format of `gs3Dg.data` is the same as that of `guess3Dg.data` (see Sec. 4.1.2), such that `gs3Dg.data` can be used as an initial guess for a new run, with for instance a different value of λ (if the grid is changed, the function must be interpolated to the new grid beforehand).

4.2 `gpoda2Dg`

This program solves the 2D GPE on a grid, corresponding to the 3D case where motion along y is frozen. Atomic units are used throughout.

4.2.1 User-supplied routines

The double precision function `potentialV(x,z)` takes as input the two double precision arguments `x` and `z`, corresponding to the spatial coordinates (x, z), and returns $V_{\rm trap}(x, z)$.

A 2D FFT routine must also be supplied. The program is set up to work with the DFFTPACK [23] transform of a real function, and can be linked directly to this library. For use of another FFT routine, please see Sec. 4.1.1.

4.2.2 Input parameters

The input parameters are read from a namelist contained in a file named `params2Dg.in`. The namelist `&params2Dg` follows the same format as the namelist `&params3Dg` presented in Sec. 4.1.2, with the omission of variables `ng_y`, `ymin`, and `ymax`. Also, the parameter `lambda` corresponds here to $g_{\rm 2D}N$ [21,22].

If the value of the input parameter `guess_from_file` is `.true.`, a file named `guess2Dg.data` must be present in the local directory. The format of the file is

similar to that of `guess3Dg.data`, presented in Sec. 4.1.2, with the exception of data corresponding to coordinate y.

4.2.3 Output files

The order parameter is written out in file `gs2Dg.data`, with each line containing the coordinates x and z, followed by $\psi(x, z)$. If the algorithm has not converged, the file will contain the function obtained at the last iteration. The format of `gs2Dg.data` is the same as that of `guess2Dg.data` (see Sec. 4.2.2), such that `gs2Dg.data` can be used as an initial guess for a new run, with for instance a different value of λ_{2D} (if the grid is changed, the function must be interpolated to the new grid beforehand).

4.3 `gpoda1Dg`

This program solves the 1D GPE on a grid, corresponding to the 3D case where motion along x and y is frozen. Atomic units are used throughout.

4.3.1 User-supplied routines

The double precision function `potentialV(z)` takes as input the double precision argument `z`, corresponding to the spatial coordinate z, and returns $V_{\rm trap}(z)$.

An FFT routine must also be supplied. The program is set up to work with the DFFTPACK [23] transform of a real function, and can be linked directly to this library. For use of another FFT routine, please see Sec. 4.1.1.

4.3.2 Input parameters

The input parameters are read from a namelist contained in a file named `params1Dg.in`. The namelist `&params1Dg` follows the same format as the namelist `&params3Dg` presented in Sec. 4.3.2, with the omission of variables `ng_x`, `ng_y`, `xmin`, `xmax`, `ymin`, and `ymax`. Also, the parameter `lambda` corresponds here to $g_{\rm 1D}N$ [20].

If the value of the input parameter `guess_from_file` is `.true.`, a file named `guess1Dg.data` must be present in the local directory. It contains the initial guess for the order parameter, and must consist in `ng_z` lines, each containing the values of the coordinate z followed by $\psi(z)$. *Note that the program does* ***not*** *check if the coordinates correspond to the grid defined by the input parameters.*

The program will simply assign the first value of ψ to the first grid point, $z_{\min}$, then the second value to the second grid point in z, and so on.

4.3.3 Output files

The order parameter is written out in file `gs1Dg.data`, with each line containing the coordinate z followed by $\psi(z)$. If the algorithm has not converged, the file will contain the function obtained at the last iteration. The format of `gs1Dg.data` is the same as that of `guess1Dg.data` (see Sec. 4.3.2), such that `gs1Dg.data` can be used as an initial guess for a new run, with for instance a different value of λ_{1D} (if the grid is changed, the function must be interpolated to the new grid beforehand).

4.4 `gpoda3Ds`

This program solves the full 3D GPE (13) using a spectral method. Note that the value of `mu` calculated is actually the rescaled $\tilde{\mu}$ defined by Eq. (16).

4.4.1 User-supplied routines

The double precision function `potentialV0(x,y,z)` takes as input the three double precision arguments `x`, `y`, and `z`, corresponding to the rescaled spatial coordinates $(\tilde{x}, \tilde{y}, \tilde{z})$, and returns $\tilde{V}_0(\tilde{x}, \tilde{y}, \tilde{z})$, defined by Eq. (14).

4.4.2 Input parameters

The input parameters are read from a namelist contained in a file named `params3Ds.in`, with the following format (the variable type is indicated in parenthesis, where dp stands for double precision):

```
&params3Ds
 lambda = nonlinearity λ̃_3D [Eq. (15)] (dp),
 wxwz = trap frequency ratio ω_x/ω_z (dp),
 wywz = trap frequency ratio ω_y/ω_z (dp),
 n_x = highest basis function in x, N_x̃ (integer),
 n_y = highest basis function in y, N_ỹ (integer),
 n_z = highest basis function in z, N_z̃ (integer),
 symmetric_x = symmetric potential in x (logical),
 symmetric_y = symmetric potential in y (logical),
 symmetric_z = symmetric potential in z (logical),
 critODA = convergence criterion for the ODA, ε_ODA (dp),
```

`critIP` = *convergence criterion for the inverse power,* $\varepsilon_{\rm IP}$ *(dp)*,
`critCG` = *convergence criterion for the conjugated gradient,* $\varepsilon_{\rm CG}$ *(dp)*,
`itMax` = *maximum number of iterations of the ODA (integer)*,
`guess_from_file` = *read initial guess from file* `guess3Ds.data`*? (logical)*
`output_grid` = *write final order parameter to file* `gs3Ds_grid.data`*? (logical)*
`&end`

The algorithm used to find the roots of the Hermite polynomial, needed for the spectral method [19], limits the acceptable highest basis function to `n` ≤ 91. The value of the parameters `symmetric` allow to reduce the size of the basis set used, for the case where the additional trapping potential V_0 [Eq. (7)] is even along any of the axes. For instance, if $V_0(x, y, z) = V_0(-x, y, z)$, setting `symmetric_x = .true.` will restrict the basis set along x to even functions $\phi(x)$ [Eq. (18)], as the order parameter will present the same parity as the trapping potential $V_{\rm trap}$. Note that in all cases the parameters `n` set the index of the highest harmonic oscillator eigenfunction used, not the number of basis functions used.

If the value of the input parameter `guess_from_file` is `.true.`, a file named `guess3Ds.data` must be present in the local directory. It contains the initial guess for the order parameter and contains lines with the values of indices i, j, and k (all integers), followed by the coefficient c_{ijk} (double precision), see Eq. (18). If an index is greater than the value of N for the corresponding spatial axis, or if its parity is not consistent with the chosen symmetry (see above), it is ignored. If a set of indices ijk appears more than once, only the last value of c_{ijk} is kept, and any c_{ijk} not specified in the file is taken to be equal to zero.

If the value of the input parameter `output_grid` is `.true.`, a second namelist will be read from the file `params3Ds.in`:

`&grid3D`
`ng_x` = *number of grid points in* $\tilde{x}$*, (integer)*,
`ng_y` = *number of grid points in* $\tilde{y}$*, (integer)*,
`ng_z` = *number of grid points in* $\tilde{z}$*, (integer)*,
`xmin` = *first point of the grid in* $\tilde{x}$ *(dp)*,
`xmax` = *last point of the grid in* $\tilde{x}$ *(dp)*,
`ymin` = *first point of the grid in* $\tilde{y}$ *(dp)*,
`ymax` = *last point of the grid in* $\tilde{y}$ *(dp)*,
`zmin` = *first point of the grid in* $\tilde{z}$ *(dp)*,
`zmax` = *last point of the grid in* $\tilde{z}$ *(dp)*
`&end`

(see next section for details on usage).

4.4.3 Output files

The order parameter is written out in file `gs3Ds.data`, with each line containing the indices i, j, and k, followed by the coefficients c_{ijk} of Eq. (18). If the algorithm has not converged, the file will contain the function obtained at the last iteration. The format of `gs3Ds.data` is the same as that of `guess3Ds.data` (see Sec. 4.4.2), such that `gs3Ds.data` can be used as an initial guess for a new run, with for instance a different value of $\tilde{\lambda}$.

If the value of the input parameter `output_grid` is `.true.`, the order parameter is also written out to the file `gs3Ds_grid.data`, with each line containing the coordinates $\tilde{x}$, $\tilde{y}$, and $\tilde{z}$, defined by the namelist `&grid3D`, followed by $\tilde{\psi}(\tilde{x}, \tilde{y}, \tilde{z})$.

4.5 `gpoda2Ds`

This program solves the a 2D GPE using a spectral method. Note that the value of `mu` calculated is actually the rescaled $\tilde{\mu}$ defined by Eq. (16).

4.5.1 User-supplied routines

The double precision function `potentialV0(x,z)` takes as input the three double precision arguments `x` and `z`, corresponding to the rescaled spatial coordinates $(\tilde{x}, \tilde{z})$, and returns $\tilde{V}_0(\tilde{x}, \tilde{z})$, defined by the 2D equivalent of Eq. (14).

4.5.2 Input parameters

The input parameters are read from a namelist contained in a file named `params2Ds.in`. The namelist `&params2Ds` follows the same format as the namelist `&params3Ds` presented in Sec. 4.4.2, with the omission of variables `wywz`, `n_y`, and `symmetry_y`. Also, the parameter `lambda` corresponds here to $\tilde{\lambda}_{2D}$ [Eq. (22)].

If the value of the input parameter `guess_from_file` is `.true.`, a file named `guess2Ds.data` must be present in the local directory. The format is the same as the file `guess3Ds.data` (Sec. 4.4.2), except that only indices i and k are present.

If the value of the input parameter `output_grid` is `.true.`, a second namelist named `&grid2D` will be read from the file `params2Ds.in`. This namelist is the same as `&grid3D` of Sec. 4.4.2, without the variables corresponding to $\tilde{y}$.

4.5.3 Output files

The order parameter is written out in file `gs2Ds.data`, with a format similar to file `gs3Ds.data` described in Sec. 4.4.3, except that only indices i and k are present. If the value of the input parameter `output_grid` is `.true.`, the order parameter is also written out to the file `gs2Ds_grid.data`, in the same manner as for file `gs3Ds_grid.data` (Sec. 4.4.3), but without the $\tilde{y}$ coordinate.

4.6 `gpoda1Ds`

This program solves the a 1D GPE using a spectral method. Note that the value of `mu` calculated is actually the rescaled $\tilde{\mu}$ defined by Eq. (16).

4.6.1 User-supplied routines

The double precision function `potentialV0(z)` takes as input the three double precision arguments `z`, corresponding to the rescaled spatial coordinate $\tilde{z}$, and returns $\tilde{V}_0(\tilde{z})$, defined by the 1D equivalent of Eq. (14).

A routine for calculating eigenvalues and eigenvectors must be supplied. The program is set up to use the LAPACK [24] routine for the eigenvalue problem for a real symmetric matrix. To use another routine, file `eigen1D.f90` has to be modified. The subroutine `eigen(n,H,eigenval,eigenvec)` takes as input the integer `n` and the double precision array `H(n,n)`. On output, the double precision real `eigenval` and the double precision array `eigenvec(n)` contain repectively the smallest eigenvalue of matrix `H` and the corresponding eigenvector.

4.6.2 Input parameters

The input parameters are read from a namelist contained in a file named `params1Ds.in`, with the following format (the variable type is indicated in parenthesis, where dp stands for double precision):

```
&params1Ds
 lambda = nonlinearity λ̃_1D [Eq. (25)] (dp),
 n = highest basis function, N (integer),
 symmetric = spatially symmetric potential? (logical),
 critODA = convergence criterion for the ODA, ε_ODA (dp),
 itMax = maximum number of iterations of the ODA (integer),
 guess_from_file = read initial guess from file guess1Ds.data? (logical)
 output_grid = write final order parameter to file gs1Ds_grid.data? (logical)
```

`&end`

See Sec. 4.4.2 for restrictions on the value of `n` and the use of `symmetric`.

If the value of the input parameter `guess_from_file` is `.true.`, a file named `guess1Ds.data` must be present in the local directory. The format is the same as the file `guess1Ds.data` (Sec. 4.4.2), except that only index k is present.

If the value of the input parameter `output_grid` is `.true.`, a second namelist named `&grid1D` will be read from the file `params1Ds.in`. This namelist is the same as `&grid1D` of Sec. 4.4.2, without the variables corresponding to $\tilde{x}$ and $\tilde{y}$.

4.6.3 Output files

The order parameter is written out in file `gs1Ds.data`, with a format similar to file `gs1Ds.data` described in Sec. 4.4.3, except that only index k is present. If the value of the input parameter `output_grid` is `.true.`, the order parameter is also written out to the file `gs1Ds_grid.data`, in the same manner as for file `gs1Ds_grid.data` (Sec. 4.4.3), but without the $\tilde{x}$ and $\tilde{y}$ coordinates.

Acknowledgments

This research was conducted in part using the resources of the High Performance Computing Center North (HPC2N).

References

[1] C. J. Pethick, H. Smith, Bose-Einstein Condensation in Dilute Gases, Cambridge University Press, Cambridge, 2002.

[2] L. Pitaevskii, S. Stringari, Bose-Einstein Condensation, Oxford University Press, Oxford, 2003.

[3] M. H. Anderson, J. R. Ensher, M. R. Matthews, C. E. Wieman, E. A. Cornell, Observation of Bose-Einstein condensation in a dilute atomic vapor, Science 269 (1995) 198.

[4] K. B. Davis, M.-O. Mewes, M. R. Andrews, N. J. van Druten, D. S. Durfee, D. M. Kurn, W. Ketterle, Bose-Einstein condensation in a gas of sodium atoms, Phys. Rev. Lett. 75 (1995) 3969–3973.

[5] C. C. Bradley, C. A. Sackett, J. J. Tollett, R. G. Hulet, Evidence of Bose-Einstein condensation in an atomic gas with attractive interactions, Phys. Rev. Lett. 75 (1995) 1687–1690.

[6] S. L. Cornish, N. R. Claussen, J. L. Roberts, E. A. Cornell, C. E. Wieman, Stable ^{85}Rb Bose-Einstein condensates with widely tunable interactions, Phys. Rev. Lett. 85 (2000) 1795–1798.

[7] T. Weber, J. Herbig, M. Mark, H.-C. Nägerl, R. Grimm, Bose-Einstein condensation of cesium, Science 299 (2003) 232–235.

[8] D. G. Fried, T. C. Killian, L. Willmann, D. Landhuis, S. C. Moss, D. Kleppner, T. J. Greytak, Bose-Einstein condensation of atomic hydrogen, Phys. Rev. Lett. 81 (1998) 3811–3814.

[9] A. Robert, O. Sirjean, A. Browaeys, J. Poupard, S. Nowak, D. Boiron, C. I. Westbrook, A. Aspect, A Bose-Einstein condensate of metastable atoms, Science 292 (2001) 461–464.

[10] F. Pereira Dos Santos, J. Léonard, J. Wang, C. J. Barrelet, F. Perales, E. Rasel, C. S. Unnikrishnan, M. Leduc, C. Cohen-Tannoudji, Bose-Einstein condensation of metastable helium, Phys. Rev. Lett. 86 (2001) 3459–3462.

[11] S. Jochim, M. Bartenstein, A. Altmeyer, G. Hendl, S. Riedl, C. Chin, J. Hecker Denschlag, R. Grimm, Bose-Einstein condensation of molecules, Science 302 (2003) 2101–2103.

[12] E. P. Gross, Structure of a quantized vortex in boson systems, Nuovo Cimento 20 (1961) 454–477.

[13] L. P. Pitaevskii, Vortex lines in an imperfect Bose gas, Sov. Phys. JETP 13 (1961) 451–454.

[14] F. Dalfovo, S. Giorgini, L. P. Pitaevskii, S. Stringari, Theory of Bose-Einstein condensation in trapped gases, Rev. Mod. Phys. 71 (1999) 463–512.

[15] S. Stenholm, Validity of the Gross-Pitaevskii equation describing bosons in a trap, Phys. Rev. A 57 (1998) 2942–2948.

[16] E. H. Lieb, R. Seiringer, J. Yngvason, Bosons in a trap: A rigourous derivation of the Gross-Pitaevskii energy functional, Phys. Rev. A 61 (2000) 043602.

[17] E. Cancès, C. Le Bris, Can we outperform the DIIS approach for electronic structure calculations?, Int. J. Quantum Chem. 79 (2000) 82–90.

[18] E. Cancès, SCF algorithms for Hartree-Fock electronic calculations, in: M. Defranceschi, C. Le Bris (Eds.), Mathematical Models and Methods for Ab Initio Quantum Chemistry, Vol. 74 of Lecture Notes in Chemistry, Springer, Berlin, 2000, pp. 17–43.

[19] C. M. Dion, E. Cancès, Spectral method for the time-dependent Gross-Pitaevskii equation with a harmonic trap, Phys. Rev. E 67 (2003) 046706.

[20] M. Olshanii, Atomic scattering in the presence of an external confinement and a gas of impenetrable bosons, Phys. Rev. Lett. 81 (1998) 938–941.

[21] D. S. Petrov, M. Holzmann, G. Shlyapnikov, Bose-Einstein condensation in quasi-2D trapped gases, Phys. Rev. Lett. 84 (2000) 2551–2555.

[22] M. D. Lee, S. A. Morgan, M. J. Davis, K. Burnett, Energy-dependent scattering and the Gross-Pitaevskii equation in two-dimensional Bose-Einstein condensates, Phys. Rev. A 65 (2002) 043617.

[23] P. N. Swarztrauber, Vectorizing the FFTs, in: G. Rodrigue (Ed.), Parallel Computations, Academic Press, New York, 1982, pp. 51–83.
URL `http://www.netlib.org/fftpack/`

[24] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, D. Sorensen, LAPACK Users' Guide, 3rd Edition, Society for Industrial and Applied Mathematics, Philadelphia, 1999.
URL `http://www.netlib.org/lapack/`

TEST RUN OUTPUT

Considering a condensate of 10^4 ^{87}Rb atoms, in a harmonic trap of frequency $\omega_x = \omega_y = \omega_z/\sqrt{8} = 2\pi \times 90$ Hz with the parameter file `params3Ds.in` as follows:

```
&params3Ds
  lambda = 368.8d0,
  wxwz = 0.353553390593d0,
  wywz = 0.353553390593d0,
  n_x = 20,
  n_y = 20,
  n_z = 20,
  symmetric_x = .true.,
  symmetric_y = .true.,
  symmetric_z = .true.,
  critODA = 1.d-8,
  critIP = 1.d-8,
  critCG = 1.d-8,
  itMax = 100,
  guess_from_file = .false.,
  output_grid = .false.
&end
```

the output will look like:

```
  GPODA3Ds

Parameters:

  omega_x / omega_z =   0.35355339E+00
  omega_y / omega_z =   0.35355339E+00
  Nonlinearity =        0.36880000E+03

  Number of basis functions:   11 x   11 x   11 =     1331
  Number of grid points:       41 x   41 x   41 =    68921
  Symmetric in x y z

  Initialization

    Compute the ground state of H_0
    Inverse Power converged in      2 iterations
     --> mu =  0.853553390593000
```

```
 Iteration   1

   Compute the ground state of H(psi_in)
   Inverse Power converged in     27 iterations
    --> mu =  2.19942774785621
   Optimal damping
     slope  -22.0705785783271
     step    0.768246751736393
     Eopt    4.08395470599574
```

[...]

```
 Iteration   66

   Compute the ground state of H(psi_in)
   Inverse Power converged in      2 iterations
    --> mu =  3.90057925938285
   Optimal damping
     slope  -1.667024296381214E-008
     step    3.537833144766566E-002
     Eopt    2.87515659549269


 Convergence achieved in  66  iterations
  --> mu =  3.90057925938285
  --> E =   2.87515659549269


 Checking self-consistency

   Inverse Power converged in      2 iterations
    --> mu =  3.90057337542902
   l1 norm of psi2out-psi2in:   0.171325E-01 for   68921 grid points
   (0.248582E-06 per grid point)
```

(Running time: 14 min on a 2.5 GHz PowerPC G5 Quad.)